Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L2	8	(James near2 Ohr).in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:28
L3	425	711/161.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:28
L4	1272	711/162.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:28
L5	40781	"709"/$.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:28
L6	27225	"711"/$.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L7	425	711/161.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L8	1272	711/162.ccls.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L9	1455	L7 or L8	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L10	191748	(backup or (back$3 adj up))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L11	337640	(restor$4 or (partial adj restor$4) or unrestor$4 or ("not" adj2 restor$3))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29

L12	1752825	map or table	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L13	191748	(backup or (back$3 adj up))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L14	337640	(restor$4 or (partial adj restor$4) or unrestor$4 or ("not" adj2 restor$3))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L15	1752825	map or table	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L16	11238	L13 and L14 and L15	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L17	605	L13 same L14 same L15	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L18	196	L13 with L14 with L15	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L19	1455	L7 or L8	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L20	196	L13 with L14 with L15	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L21	31	L20 and L19	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29

L22	605	L13 same L14 same L15	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L23	82	L22 and L19	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L24	297710	SAN	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L25	31	L20 and L19	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L26	297710	SAN	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L27	11	L25 AND L26	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L28	82	L22 and L19	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L29	26	L28 AND L26	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L30	276	(unrestor$4 or ("not" near3 restor$3))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L31	26	L28 AND L26	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29

L32	276	(unrestor$4 or ("not" near3 restor$3))	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L33	0	L31 and L32	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L34	11	L25 AND L26	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L35	0	L34 and L32	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L36	5	L19 and L32	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29
L37	70	(restor$4 adj2 remainder$2)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/12/12 11:29

Terms used
data restoration file server data block granularity backup backing up restore restoration

Found **301** of **167,655**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Efficient distributed backup with delta compression

Randal C. Burns, Darrell D. E. Long

November 1997 **Proceedings of the fifth workshop on I/O in parallel and distributed systems**

Publisher: ACM Press

Full text available: pdf(1.37 MB) Additional Information: full citation, references, citings, index terms

2 Peer-to-peer infrastructure: Pastiche: making backup cheap and easy

Landon P. Cox, Christopher D. Murray, Brian D. Noble

December 2002 **ACM SIGOPS Operating Systems Review**, Volume 36 Issue SI

Publisher: ACM Press

Full text available: pdf(1.65 MB) Additional Information: full citation, abstract, references

Backup is cumbersome and expensive. Individual users almost never back up their data, and backup is a significant cost in large organizations. This paper presents *Pastiche*, a simple and inexpensive backup system. Pastiche exploits excess disk capacity to perform peer-to-peer backup with no administrative costs. Each node minimizes storage overhead by selecting peers that share a significant amount of data. It is easy for common installations to find suitable peers, and peers with high ove ...

3 Frangipani: a scalable distributed file system

Chandramohan A. Thekkath, Timothy Mann, Edward K. Lee

October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles SOSP '97**, Volume 31 Issue 5

Publisher: ACM Press

Full text available: pdf(2.20 MB) Additional Information: full citation, references, citings, index terms

4 File servers for network-based distributed systems

Liba Svobodova

December 1984 **ACM Computing Surveys (CSUR)**, Volume 16 Issue 4

Publisher: ACM Press

Full text available: Additional Information: full citation, references, citings, index terms,

pdf(4.23 MB) review

5 A coherent distributed file cache with directory write-behind

Timothy Mann, Andrew Birrell, Andy Hisgen, Charles Jerian, Garret Swart
May 1994 **ACM Transactions on Computer Systems (TOCS)**, Volume 12 Issue 2
Publisher: ACM Press

Full text available: pdf(3.21 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Extensive caching is a key feature of the Echo distributed file system. Echo client machines maintain coherent caches of file and directory data and properties, with write-behind (delayed write-back) of all cached information. Echo specifies ordering constraints on this write-behind, enabling applications to store and maintain consistent data structures in the file system even when crashes or network faults prevent some writes from being completed. In this paper we describe ...

Keywords: coherence, file caching, write-behind

6 Distributed file systems: concepts and examples

Eliezer Levy, Abraham Silberschatz
December 1990 **ACM Computing Surveys (CSUR)**, Volume 22 Issue 4
Publisher: ACM Press

Full text available: pdf(5.33 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The purpose of a distributed file system (DFS) is to allow users of physically distributed computers to share data and storage resources by using a common file system. A typical configuration for a DFS is a collection of workstations and mainframes connected by a local area network (LAN). A DFS is implemented as part of the operating system of each of the connected computers. This paper establishes a viewpoint that emphasizes the dispersed structure and decentralization of both data and con ...

7 Distributed, object-based programming systems

Roger S. Chin, Samuel T. Chanson
March 1991 **ACM Computing Surveys (CSUR)**, Volume 23 Issue 1
Publisher: ACM Press

Full text available: pdf(2.97 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The development of distributed operating systems and object-based programming languages makes possible an environment in which programs consisting of a set of interacting modules, or objects, may execute concurrently on a collection of loosely coupled processors. An object-based programming language encourages a methodology for designing and creating a program as a set of autonomous components, whereas a distributed operating system permits a collection of workstations or personal computers ...

Keywords: capability scheme, distributed operating systems, error recovery, method invocation, nested transaction, object model, object reliability, object-based programming languages, processor allocation, resource management, synchronization, transaction

8 A cryptographic file system for UNIX

Matt Blaze

December 1993 **Proceedings of the 1st ACM conference on Computer and communications security**
Publisher: ACM Press

Full text available: pdf(955.62 KB) Additional Information: full citation, abstract, references, citings, index terms

Although cryptographic techniques are playing an increasingly important role in modern computing system security, user-level tools for encrypting file data are cumbersome and suffer from a number of inherent vulnerabilities. The Cryptographic File System (CFS) pushes encryption services into the file system itself. CFS supports secure storage at the system level through a standard Unix file system interface to encrypted files. Users associate a cryptographic key with the directories ...

9 FS2: dynamic data replication in free disk space for improving disk performance and energy consumption
Hai Huang, Wanda Hung, Kang G. Shin
October 2005 **ACM SIGOPS Operating Systems Review , Proceedings of the twentieth ACM symposium on Operating systems principles SOSP '05**, Volume 39 Issue 5
Publisher: ACM Press
Full text available: pdf(542.63 KB) Additional Information: full citation, abstract, references, index terms

Disk performance is increasingly limited by its head positioning latencies, i.e., seek time and rotational delay. To reduce the head positioning latencies, we propose a novel technique that *dynamically* places copies of data in file system's *free blocks* according to the disk access patterns observed at runtime. As one or more replicas can now be accessed in addition to their original data block, choosing the "nearest" replica that provides fastest access can significantly improve pe ...

Keywords: data replication, disk layout reorganization, dynamic file system, free disk space

10 System R: relational approach to database management
M. M. Astrahan, M. W. Blasgen, D. D. Chamberlin, K. P. Eswaran, J. N. Gray, P. P. Griffiths, W. F. King, R. A. Lorie, P. R. McJones, J. W. Mehl, G. R. Putzolu, I. L. Traiger, B. W. Wade, V. Watson
June 1976 **ACM Transactions on Database Systems (TODS)**, Volume 1 Issue 2
Publisher: ACM Press

Full text available: pdf(3.18 MB) Additional Information: full citation, abstract, references, citings, index terms

System R is a database management system which provides a high level relational data interface. The systems provides a high level of data independence by isolating the end user as much as possible from underlying storage structures. The system permits definition of a variety of relational views on common underlying data. Data control features are provided, including authorization, integrity assertions, triggered transactions, a logging and recovery subsystem, and facilities for maintaining ...

Keywords: authorization, data structures, database, index structures, locking, nonprocedural language, recovery, relational model

11 Assembly instruction level reverse execution for debugging
Tankut Akgul, Vincent J. Mooney III
April 2004 **ACM Transactions on Software Engineering and Methodology (TOSEM)**, Volume 13 Issue 2
Publisher: ACM Press

Full text available: pdf(1.18 MB) Additional Information: full citation, abstract, references, index terms

Assembly instruction level reverse execution provides a programmer with the ability to return a program to a previous state in its execution history via execution of a "reverse program." The ability to execute a program in reverse is advantageous for shortening software development time. Conventional techniques for recovering a state rely on saving the state into a record before the state is destroyed. However, state-saving causes significant memory and time overheads during forward execution.Th ...

Keywords: Debugging, reverse code generation, reverse execution

12 Improving storage system availability with D-GRAID

Muthian Sivathanu, Vijayan Prabhakaran, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau

May 2005 **ACM Transactions on Storage (TOS)**, Volume 1 Issue 2

Publisher: ACM Press

Full text available: pdf(700.30 KB) Additional Information: full citation, abstract, references, index terms

We present the design, implementation, and evaluation of D-GRAID, a gracefully degrading and quickly recovering RAID storage array. D-GRAID ensures that most files within the file system remain available even when an unexpectedly high number of faults occur. D-GRAID achieves high availability through aggressive replication of semantically critical data, and fault-isolated placement of logically related data. D-GRAID also recovers from failures quickly, restoring only live file system data to a h ...

Keywords: Block-based storage, Disk array, RAID, fault isolation, file systems, smart disks

13 Log files: an extended file service exploiting write-once storage

R. Finlayson, D. Cheriton

November 1987 **ACM SIGOPS Operating Systems Review , Proceedings of the eleventh ACM Symposium on Operating systems principles SOSP '87**, Volume 21 Issue 5

Publisher: ACM Press

Full text available: pdf(1.07 MB) Additional Information: full citation, abstract, references, citings, index terms

A log service provides efficient storage and retrieval of data that is written sequentially (append-only) and not subsequently modified. Application programs and subsystems use log services for recovery, to record security audit trails, and for performance monitoring. Ideally, a log service should accommodate very large, long-lived logs, and provide efficient retrieval and low space overhead.In this paper, we describe the design and implementation of the Clio log service. Clio pr ...

14 Scale and performance in a distributed file system

John H. Howard, Michael L. Kazar, Sherri G. Menees, David A. Nichols, M. Satyanarayanan, Robert N. Sidebotham, Michael J. West

February 1988 **ACM Transactions on Computer Systems (TOCS)**, Volume 6 Issue 1

Publisher: ACM Press

Full text available: pdf(2.38 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The Andrew File System is a location-transparent distributed tile system that will eventually span more than 5000 workstations at Carnegie Mellon University. Large scale affects performance and complicates system operation. In this paper we present observations of a prototype implementation, motivate changes in the areas of cache

validation, server process structure, name translation, and low-level storage representation, and quantitatively demonstrate Andrews ability to scale gracefully. W ...

15 Disaster recovery techniques for database systems

Manhoi Choy, Hong Va Leong, Man Hon Wong
November 2000 **Communications of the ACM**
Publisher: ACM Press
Full text available: pdf(412.04 KB) Additional Information: full citation, references, index terms

16 A Self-Organizing Storage Cluster for Parallel Data-Intensive Applications

Hong Tang, Aziz Gulbeden, Jingyu Zhou, William Strathearn, Tao Yang, Lingkun Chu
November 2004 **Proceedings of the 2004 ACM/IEEE conference on Supercomputing**
Publisher: IEEE Computer Society
Full text available: pdf(330.26 KB) Additional Information: full citation, abstract

Cluster-based storage systems are popular for data-intensive applications and it is desirable yet challenging to provide incremental expansion and high availability while achieving scalability and strong consistency. This paper presents the design and implementation of a self-organizing storage cluster called Sorrento, which targets data-intensive workload with highly parallel requests and low write-sharing patterns. Sorrento automatically adapts to storage node joins and departures, and the sys ...

17 Algorithms and data structures for flash memories

Eran Gal, Sivan Toledo
June 2005 **ACM Computing Surveys (CSUR)**, Volume 37 Issue 2
Publisher: ACM Press
Full text available: pdf(343.39 KB) Additional Information: full citation, abstract, references, index terms

Flash memory is a type of electrically-erasable programmable read-only memory (EEPROM). Because flash memories are nonvolatile and relatively dense, they are now used to store files and other persistent objects in handheld computers, mobile phones, digital cameras, portable music players, and many other computer systems in which magnetic disks are inappropriate. Flash, like earlier EEPROM devices, suffers from two limitations. First, bits can only be cleared by erasing a large block of memory. S ...

Keywords: EEPROM memory, Flash memory, wear leveling

18 Programming languages for distributed computing systems

Henri E. Bal, Jennifer G. Steiner, Andrew S. Tanenbaum
September 1989 **ACM Computing Surveys (CSUR)**, Volume 21 Issue 3
Publisher: ACM Press
Full text available: pdf(6.50 MB) Additional Information: full citation, abstract, references, citings, index terms, review

When distributed systems first appeared, they were programmed in traditional sequential languages, usually with the addition of a few library procedures for sending and receiving messages. As distributed applications became more commonplace and more sophisticated, this ad hoc approach became less satisfactory. Researchers all over the world began designing new programming languages specifically for implementing distributed applications. These languages and their history, their underlying pr ...

19 Ext3cow: a time-shifting file system for regulatory compliance

Zachary Peterson, Randal Burns

May 2005 **ACM Transactions on Storage (TOS)**, Volume 1 Issue 2
Publisher: ACM Press
Full text available: pdf(443.01 KB) Additional Information: full citation, abstract, references, index terms

The ext3cow file system, built on the popular ext3 file system, provides an open-source file versioning and snapshot platform for compliance with the versioning and audtitability requirements of recent electronic record retention legislation. Ext3cow provides a *time-shifting* interface that permits a real-time and continuous view of data in the past. Time-shifting does not pollute the file system namespace nor require snapshots to be mounted as a separate file system. Further, ext3cow is i ...

Keywords: Versioning file systems, copy-on-write

20 The Alpine file system

M. R. Brown, K. N. Kolling, E. A. Taft
November 1985 **ACM Transactions on Computer Systems (TOCS)**, Volume 3 Issue 4
Publisher: ACM Press
Full text available: pdf(2.95 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Alpine is a file system that supports atomic transactions and is designed to operate as a service on a computer network. Alpine's primary purpose is to store files that represent databases. An important secondary goal is to store ordinary files representing documents, program modules, and the like. Unlike other file servers described in the literature, Alpine uses a log-based technique to implement atomic file update. Another unusual aspect of Alpine is that it performs all commu ...

Mon, 12 Dec 2005, 11:38:50 AM EST

Edit an existing query or compose a new query in the Search Query Display.

Select a search number (#) to:

- Add a query to the Search Query Display
- Combine search queries using AND, OR, or NOT
- Delete a search
- Run a search

Search Query Display

Recent Search Queries

#1 ((data backup)<in>metadata)

#2 ((data restoration)<in>metadata)

#3 (data block granularity<IN>metadata)

#4 (file server<IN>metadata)

#5 (file server<in>metadata) and (backup)

#6 (file server<in>metadata) and (backup) and restore

#7 (file server<in>metadata) and (backup) and restore and (block level)